PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 17/30	A1	(11) International Publication Number: WO 00/23915
		(43) International Publication Date: 27 April 2000 (27.04.00)

(21) International Application Number: PCT/SG99/00100

(22) International Filing Date: 6 October 1999 (06.10.99)

(30) Priority Data:
9803829-2 16 October 1998 (16.10.98) SG

(71) Applicant *(for all designated States except US)*: KENT RIDGE DIGITAL LABS [SG/SG]; 21 Heng Mui Keng Terrace, Singapore 119613 (SG).

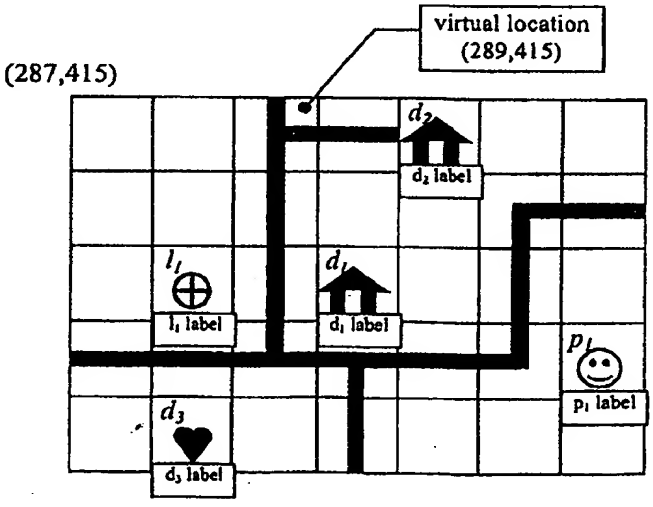
(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: LOO, Peing, Ling [SG/SG]; 75B Everitt Road, Singapore 428618 (SG). CHEW, Sor, Kuan [SG/SG]; Blk 336, Clementi Avenue 2 #16-34, Singapore 120336 (SG). CHUA, Beng, Choon [SG/SG]; Blk 270, Toh Guan Road #15-97, Singapore 600270 (SG).

(74) Agent: GREENE-KELLY, James, Patrick; Lloyd Wise, Tanjong Pagar, P.O. Box 636, Singapore 910816 (SG).

(81) Designated States: JP, SG, US, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: A VIRTUAL SPACE



(57) Abstract

A virtual space is disclosed formed as a cartesian coordinate grid which defines cells at the intersections of the grid each of which can be associated with at least one object. The objects can be documents such as web pages, avatars of users or navigation aids allowing movement between cells. The user can explore the virtual space by moving his/her avatar between cells and interacting with objects in the cells. The grid can be distributed over a plurality of servers in an internet environment, each server being responsible for a region of the grid and a user can travel across boundaries between the regions, the user communicating with the respective server as the user enters its region. The virtual space may be implemented using the viewing metaphor of a town in which objects are associated with representations which, in the mind of the user, have broad functional similarities with the objects represented.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

A VIRTUAL SPACE

FIELD AND BACKGROUND OF THE INVENTION

This invention relates to a virtual space implementable on a computer system.

Virtual reality programs have been proposed, such as those sold under the trademarks VISCAPE and ACTIVE WORLD. Such programs allow the construction of a virtual space, such as a virtual world, for uses such as simulation and communication, for example.

It is a disadvantage of existing programs for the construction of a virtual space that the programs tend to require extensive computer power to function, particularly due to the representation of 3D graphics, and are thus slow and clumsy to use. Such programs also do not let the user have a consistent sense of direction, resulting in disorientation during navigation. This problem is principally caused by the features used for travelling between regions of the virtual space. Such features, generally called portals or doors allow the user to jump between different points in the virtual space at either side of the portal. On arriving at the portal's destination, the user has little idea of to where he or she has moved in the virtual space, relative to where the user has come from, thus giving the virtual space an abstract appearance to the user.

It is an object of the invention to provide an improved virtual space.

SUMMARY OF THE INVENTION

According to the invention, there is provided a virtual space extending over a plurality of servers, the space comprising a grid defining a plurality of cells at coordinate points of the grid and means for associating each cell with at least one object and the grid comprises a plurality of regions, each region being disposed on a respective server.

Each server may have jurisdiction over its region and each region may be connected to at least one other region at a common boundary, the connected regions preferably being coordinately contiguous.

An object may comprise a document, may connect different parts of the virtual space, may reference the coordinates of another cell, may define a user of the virtual space or may reference another object in the virtual space.

Each object and the location thereof may be defined in an object table and the object table may be implemented as a sparse matrix, each server preferably having a said object table defining objects existing in cells in the region of the server.

The virtual space may further have means for referencing a plurality of objects together and said means may comprise a defined array of said plurality of objects. At least one of said plurality of objects may reference a further plurality of objects.

The virtual space may further comprise a user table which identifies users of the space and each server may have a user table identifying users having home locations in the region of the server. Each entry in the or each user table may include an identification of the user in the table, a name for the user, a password associated with the user and a contact address which may be an E-Mail address.

The virtual space may further comprise at least one group table, identifying a group of users of the space and each server may have a group table identifying a group of users having home locations in the region of the server. Each entry in the or each group table may include a definition of a group and an identification of a user that is part of that group.

The virtual space may further comprise at least one table of access control zones, each zone having access requirements associated therewith and each server may have a said table of access zones, each zone being disposed within the region of each server. A table of badges may further be provided, each badge having access rights associated therewith, access

to a zone being granted when the access rights of a holder of a badge meet the access requirements of a zone and a rights table of users and the badges held by each user. Each server may have a rights table containing badges held by users having home locations in the region of the server, the rights table of each server preferably containing badges for zones in the region of the server. The rights table of each server may also contain badges held by users having home locations in regions of other servers.

The virtual space may further comprise means for displaying a portion of the grid and representations of objects contained in cells in said portion, the portion containing the virtual location of the user. Preferably, the virtual location of the user is at the centre of the displayed portion. Alternatively, the portions may be non-overlapping parts of the grid and a corner of the displayed portion preferably has grid coordinates which are integer divisible by the cell dimensions of the portion. When a portion extends over more than one region, the display means preferably obtains information concerning the objects within the part of each region to be displayed from the region's server and displays the parts together as a single display.

Objects that reference another object in the virtual space may be stored in a table of referring objects and corresponding referred objects, each server preferably having a table of referring objects and corresponding referred

objects in respect of referring objects within its region.

The virtual space may further comprise means for updating references to and views of objects, each server preferably updating references and views within its region.

The virtual space may further comprise means for changing the virtual location of the user preferably by using an object to change the virtual location and such an object may be of a type mentioned above. The means may also or in the alternative comprise means for moving to the location of a selected object by searching for objects having at least one attribute and selecting the object from the search results or may comprise means for specifying a zone, displaying references to objects associated with cells within that zone, selecting a referenced object and moving to the virtual location of that object. The zone may include the virtual position of the user and may be centred on the virtual position of the user. The zone may be of a regular geometric virtual shape. The specifying means may specify a plurality of zones of expanding size and may display object references within one zone before displaying object references within the next larger zone.

The virtual space may further comprise means for generating default objects in cells of the grid and this may be achieved using a populating function of the form:

$$f(x,y) \rightarrow \{0,1\}$$

where x and y are the coordinates of the cell in which the default object is to be disposed.

The virtual space preferably has a viewing metaphor which may be that of a community, most preferably that of a town in which a document object is represented by a building, an object connecting different parts of the virtual space by a road, by public transport for an object which references the coordinates of another cell and by an avatar of the user for a user-defining object.

Preferably, the plurality of servers are interconnected via the internet.

The described embodiment of the invention provides a virtual space formed as a grid, for example a Cartesian coordinate grid. Preferably the grid extends across several host computers termed servers and each server is responsible for at least one region of the grid termed a jurisdiction. Links between jurisdictions on different servers occur at the boundaries of the jurisdictions. Thus, when the user navigates from one jurisdiction to the next, a feeling is provided of continuity as if the user was navigating on a physical landscape. The described embodiment of the invention thus provides a virtual space, which provides the user with a direction-based environment to which the user can relate more easily.

In a preferred example, the virtual space is represented at the user interface using a community metaphor, in particular a town metaphor, termed "Common Town". In this representation, the virtual space is represented as a common piece of land across different servers, with each server managing a pre-assigned piece of land (the jurisdiction of that server). Users moving across the boundaries between server jurisdictions are transferred to the appropriate server that manages the appropriate piece of land. Each grid square of the land can be occupied by icons of various kinds of objects, such as buildings and roads, in addition to representations of the users. The kinds of icons relate to their function. For example, a building can represent a document, the content of which is displayed when the user accesses it and/or can represent a meeting point, such as a newsgroup. Movement corridors represented as roads can be used to travel across the virtual space.

The selection of metaphor, icons and types of objects is, however, essentially arbitrary, and is governed by the purpose to which the virtual space is put.

BRIEF DESCRIPTION OF THE DRAWINGS

An embodiment of the invention will now be described with reference to the accompanying drawings in which:

Fig. 1 is a schematic diagram of the structure of the virtual

space of the embodiment of the invention;

Fig. 2 is a schematic diagram illustrating an aggregate object;

Fig. 3 is a schematic diagram of a display of the virtual space;

Fig. 4 illustrates an isometric and more detailed display;

Fig. 5 is a schematic diagram illustrating a display provided by information from three servers.

Fig. 6 illustrates the display of a stand-in object; and

Fig. 7 shows a view of the grid illustrating regions of jurisdiction of three servers.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

In explaining the described embodiment of the virtual space of the invention, reference will be made to how the space is constructed on a computer system. An example of a user interface of the virtual space in which a "town" metaphor is adopted will then be described.

The construction of the virtual space will first be described with reference to its structure and the types of objects associated with cells of the virtual space (Section 1); the definition of users and the control of access to the virtual space will be described in Section 2; the display and viewing of the virtual space by a user on a client's computer system is described in Section 3; the techniques for a user to navigate within the virtual space are described in Section 4;

techniques for defining default objects are described in Section 5; the Application Program Interface (API) for use with the virtual space is described in Section 6 and the use of a metaphor for the user interface of the virtual space is described in Section 7, with the specific example of a user interface using a town metaphor being described in Section 8.

1. Virtual Space Structure and Object Types

1.1 The Structure of the Virtual Space

The virtual space is defined as a conceptual grid with reference to a square cell given the coordinates (0,0) which are values in an x and y axis two-dimensional Cartesian coordinate system. This is known as the "origin". Cells of the same size extend throughout the virtual space in both the horizontal (x) and vertical (y) direction. Each cell has a pair of coordinates in the form of (CX, CY), which define the position of the cell relative to the origin. This is illustrated in Figure 1.

Each cell may be empty or contain one or more objects. The computer system defines an OBJECTS table to store the objects in the cells, the OBJECTS table thus representing the grid by specifying the conceptual coordinate location of all objects relative to the origin. Each entry of this table holds an object and has the following elements:

(OID, CX, CY, LAYER, PICT, TYPE, LABEL, BEHAVE, TSTAMP, DREF)

Where the elements are:

OID: An identification that uniquely identifies the object in the OBJECTS table.

CX: The horizontal coordinate of the cell.

CY: The vertical coordinate of the cell.

LAYER: The stacking order of the object in the cell. This is used to uniquely identify multiple objects at a cell location, so that if there are n objects in a cell, these are assigned layer numbers 1, 2....

PICT: The visual representation or the "look" of the object, this can be the location of an icon, an animation (a series of icons) or a 3D model etc.

TYPE: The type of object in the cell.

LABEL: The label to be displayed together with the object.

TSTAMP: The time and date the object was created.

DREF: The reference or address to data associated with the object. This can be a reference to a computer document such as a data file or an Internet link. For some object types, DREF is a CX and CY pair. For some object types, this element is not used as is mentioned in Section 1.2 below.

With this structure, the OBJECTS table is able to be implemented as a sparse matrix, providing space efficiency since storage is not needed for empty cells. Due to the

sparse matrix implementation, the OBJECTS table can hold cells at any distance from the origin (0,0). Thus, the virtual space defined in this way can be extended effectively indefinitely outward in both x and y directions.

A single computer system can define the entire virtual space. However, due to physical constraints (such as storage, data transmission speed and number of users), it is more useful to design and implement a distributed architecture which defines this single virtual space among a number of computer systems hereinafter referred to as servers.

Each server controls a region of this virtual space, termed "a region of jurisdiction". It is defined as a collection of rectangles each takes the form of:

(BX, BY, EX, EY)

... where the B elements are the x and y coordinates of the top-left corner cell and the E elements are the x and y coordinates of the bottom-right corner cell of the region. This collection of rectangles defines a region of jurisdiction that may be irregular in shape and not necessarily contiguous.

The regions of jurisdiction defined by the plurality of servers must be mutually exclusive, that is not overlapping so that no one cell is controlled by more than a server.

This is illustrated in Figure 7 in which regions of three servers S_1 - S_3 are shown. Each server has its own OBJECTS table and the entries in this table only describe cells within the jurisdiction of that server. The entire virtual space need not be exhaustively defined. Certain cells can be free from the control of any server, but such cells will not be accessible to users.

Servers can be added over time. Since the virtual space has no boundaries by definition, any number of servers can be added.

The addition of servers is managed by a computer program termed a Land Bureau which may be separately provided on one of the servers or on a separate computer system. The Land Bureau maintains two tables termed SREGIONS and SERVERS having the following elements:

(SID, BX, BY, EX, EY) ----- SREGIONS

(SID, SNAME, SADDR)----- SERVERS

Where the elements are :

SID: A unique identification for the server.

BX: The x coordinate of the top-left corner cell of the region.

BY: The y coordinate of the top-left corner cell of the region.

EX: The x coordinate of the bottom-right corner cell of

the region.

EY: The y coordinate of the bottom-right corner cell of the region.

SNAME: A unique name of the server.

SADDR: The network address of the server.

There can be several SREGIONS table entries with the same SID (thus defining the irregular shaped region of jurisdiction of a server) but there must be one entry in the SERVERS table to describe a server.

The Land Bureau transmits these two tables to the servers periodically. Thus, each server has a read-only copy of each of these tables but modification is not allowed. These tables enable a server to:

1. Determine whether a cell is within its region of jurisdiction and allowing the corresponding entry in the OBJECTS table to be created, modified or deleted.

2. If a cell falls outside of its jurisdiction, determine which server's jurisdiction it falls into, and inform this target server to perform the necessary operations instead.

Each server uses an inclusion test on the entries in the SREGIONS table to accomplish this test of jurisdiction. For this test, taking a point (x, y), it is within the region R:(sid, bx, by, ex, ey) if $bx \leq x \leq es$ and $by \leq y \leq ey$. A

server's jurisdiction is specified by a set of regions (ie. those which SID is same as the server's), if a point is within any of these regions, it is said to be within the server's jurisdiction.

1.2 Object Types

Objects are said to occupy cells in this virtual space if they are defined in the OBJECTS table. Some possible object types will now be described, although it will be appreciated that the possibilities are unlimited and thus not all types of objects can be completely described here.

Object type **OBJDOC**: The simplest type of object is termed OBJDOC. The DREF element in the object table references to a computer document.

Object type **OBJPATH**: The type OBJPATH defines objects which connect different parts of the virtual space and are useful for navigation. OBJPATH objects do not have a DREF element.

Object type **OBJLINK**: The type OBJLINK defines objects which are useful from navigating instantaneously from one cell in the virtual space to another (i.e. a "destination"). The DREF element references the destinations cell's coordinates.

Object type **OBJLIFE**: The type OBJLIFE defines objects which are avatars of users. Avatars are net-representations of

users and each user has an OBJLIFE object which represents the user. The DREF element references an I.D. for the user such as an E-mail address.

Object type **OBJSTIN**: The type OBJSTIN defines objects which are "stand-ins" of other objects (targets). These stand-in objects mirror and behave exactly as the objects for which they stand-in. The DREF element contain the OID of the target.

Object type **OBJSIGN**: This type of object provides a marker in the virtual space. No other type of object is referenced by the marker, which is for searching purposes. This type of object is also used as a visual cue for users. The DREF element is consequently not used.

Object type **OBJPTR**: This object type is similar to an OBJSIGN object but, instead of marking a particular point in the virtual space, the object informs the user, as a visual cue, of information concerning adjacent regions of the virtual space to that actually being viewed. Like an OBJSIGN object, OBJPTR is not directly associated with any other object and the DREF element is not used.

Objects with DREF elements in the form of (CX, CY) automatically reference across servers since a single uniform coordinate system is used. An example is objects of OBJLINK type. Objects such as OBJSTIN, which references other

objects using the OID element need additional information instead when referencing "foreign objects". The DREF element thus contains the (SID, OID) pair when referencing across servers (SID is the unique server identification).

1.3 Aggregate Objects

The virtual space can contain arrays of objects. An array of objects is a list of contiguous cells which contain objects. Although not needed in principle, in this embodiment, the convention is adopted that arrays only extend from left to right (i.e. in the direction of increasing x value). An array is terminated by an empty cell.

An array of objects has no inherent semantic by itself. An "aggregate object" makes use of an array of objects by referencing the first member ("head of array") using the DREF element. In Figure 2 below, objects o1, o2, o3, ... on occupy contiguous cells and form an array. The aggregate object A references it by "pointing" to the start of the array with its DREF element.

There are many possible types of aggregate objects and applications can define them when needed. For example an array of OBJLIFE objects is defined as an OBJALIFE aggregate object. OBJALIFE aggregate objects, being net representations of users, could be used as mailing lists for an electronic mail application. Similarly, arrays of OBJLINK objects can

form an OBJALINK aggregate object which specify a plurality of locations to which the user can move.

The interpretation of arrays of objects is left to the aggregate objects and the applications which define them. Array definitions can be "nested", that is, members of an array can be arrays themselves.

2. Users and Access

2.1 Users and Groups

2.1.1 Users

As described above, OBJLIFE objects are used to represent users in the virtual space. Besides these objects, there is a table USERS on each server that stores information of users and is used for controlling access. Each entry of USERS includes the following elements:

(UID, UNAME, PASSWD, UTYPE, EMAIL, UOBJREF)---- USERS

Where the elements are:

UID: A unique identification of the user entry in the USERS table.

UNAME: A unique name of the user in this server.

PASSWD: An encrypted password the user use to verify his identify (during sign-in to the virtual space).

UTYPE: The type of user. Some examples are UUSER which is an individual user and UGROUP which is a group as discussed below.

EMAIL: The user's email address.

UOBJREF: The OID of the user's avatar object (usually of the OBJLIFE type, except, when UTYPE element contains the type UGROUP. That is, if the entry is a group description as described below.

2.1.2 Groups

Groups define a plurality of users.

Groups make assigning of badges (see below) to users easier. For each new user that registers with the server, it might be laborious to assign the user with the badges for the zones the user is allowed to access. With the concept and implementation of groups, if only needs to be decided what groups the user wishes to belong to and make the membership assignment, the server then making the computation to assign the user with the appropriate badges. As an example: student John is a member of (groups) Oakey High School, Class 3C, Math Society and Jellyfish (pop-band); assigning him as member to these groups will grant him the necessary badges for different zones in the virtual space (which are the zones shared by these groups) without having to assign individual user badges to him.

Groups are represented as "virtual users". They are users who have entries in the USERS table which UTYPE element contains the type UGROUP.

Groups can have members who are other users (or groups). A table GMEMBERS stored this member information. Each entry in the table includes the following elements:

(GREF, UREF)

Where the elements are:

GREF: The definition of a group, being a reference to a group entry in the USERS table. It must be a UID of an entry with a UGROUP type.

UREF: The definition of a user that is part of the group defines by GREF by reference to an entry in the USERS table. This can be an UID of either an UUSER or UGROUP type.

Each server ensures the integrity of the GMEMBERS table by ensuring that it contains valid membership information. This is by done checking that there are no self-referential or recursive definitions of membership.

2.2 Zones and badges

The concepts of "zones" and "badges" are used as the basis of the access control mechanism in the virtual space. A zone

defines an exclusive region of the virtual space and assigns it with default access types. Tables termed ZREGIONS and ZONES store information of zones. Each entry has the following components:

(ZID, BX, BY, EX, EY)---- ZREGIONS

(ZID, ACSET)---- ZONES

Where the elements are:

ZID: A unique identification of the zone.

BX: The x coordinate of the top-left corner cell of the zone.

BY: The y coordinate of the top-left corner cell of the zone.

EX: The x coordinate of the bottom-right corner cell of the zone.

EY: The y coordinate of the bottom-right corner cell of the zone.

ACSET: The default set of access types allowed to any user on the zone. These include, but are not limited to, ACADMIN, ACREAD, ACWRITE, ACMODIFY, and ACDELETE which allow the user the administrate, read, write, modify or delete contents of cells in the zone.

ZID does not have to be unique in the ZREGIONS table but there must be only one entry for each ZID in ZONES table. The area defined by each zone can be an irregularly-shaped region made up of several rectangles. Zones must be mutually

exclusive (non-overlapping). A server can only define zones within its jurisdiction.

Badges define additional accesses to zones. A badge defines, always in relation to a zone, a set of access types allowed to computer software applications. The table BADGES on a server stores the information of the badges for zones in that server's jurisdiction. Each entry has the following elements:

(BID, ZREF, ACSET)---- BADGES

Where the elements are:

BID: An identification of the badge. Unique in the table.

ZREF: The ZID of the zone with which the badge is associated.

ACSET: The additional set of access types allowed on the zone to users who possess the badge.

Each server maintains a UBADGES table which stores the information of to whom the badges are granted. Each entry of this table has the following elements:

(UREF, BREF)---- UBADGES

Where the elements are:

UREF: The UID (from the USERS table) of the user or

group.

BREF: The BID (from the BADGES table) of the badge.

2.3 Access Control of Objects

Access control can be defined for individual objects rather than zones. This is done through the use of an OCONTROL table on each server which has the following elements:

(OREF, UREF, ACSET)---- OCONTROL

Where the elements are:

OREF: The OID of a local object.

UREF: The UID of a user to be granted the access right.

ACSET: The additional set of access types allowed on the object to the user specified in UREF.

2.4 Access Right resolution

These steps are followed in order to obtain a user's access right to a cell by the server having the cell within its jurisdiction:

1. Obtain the zone, z, to which the cell belongs.
2. Obtain the set of all badges Bz associated with z.
3. Obtain the set of all badges Bu granted to the user directly or indirectly through group membership.
4. Intersect Bz and Bu to get B:{b1, b2, ...bn} - the set of

badges relevant to zone z and the user.

5. The user's access right to the cell is the set Ac, the union of all the ACSET elements of z, b1, b2, ... and bn.

6. If there is an object in the cell, the OCONTROL table is also looked up to obtain Ao, the set of access types associated with the object. The user's access right to the object is the set A, the union of Ac and Ao.

2.5 Users and badges across servers

Users belong to a server are known as "local users" to the server, otherwise they are "foreign users". Likewise, there is the "local server" and "foreign servers". A server refers to a foreign user by the pair (SID, UID) where SID is the server identification and UID is the user identification. A foreign user can be of type UGROUP, making it a "foreign group".

In the GMEMBERS table, the UREF element can be a reference to a foreign user (or group). This means that a group can have foreign users (or groups) as its members. However, GREF cannot be foreign.

Badges are defined in relation to zones which must always be within a server's jurisdiction. As such, by definition, a server cannot define a badge outside of its jurisdiction in its BADGES table. However, users of a server can be granted badges by a foreign server. This is necessary since we

should not restrict users to access only the space governed by their own servers. In the UBADGES table, UREF can be foreign, making it possible for foreign users to obtain badges.

3. Views and Displays

3.1 Displaying the virtual space

The virtual space is a large 2D space. A computer monitor can only display a portion of this virtual space at a time in the form of a rectangular region of cells. This region is known as a "display". The display is determined by the user's "virtual location" which is defined by the coordinates of a cell. There are two possible schemes to determine the dimension of the display:

1. The cell defining the user virtual location is always the centre of the display. As a result, the display changes as often as the virtual location.

2. The display is always chosen such that the cell at the top-left corner has coordinates integer divisible by the dimension of the display. If, for example, the virtual location is (289, 415) and the display has size 7 by 5 cells, the cell at the top-left corner is (287, 415). This scheme is less processing and network intensive because the display is "snapped" into non-overlapping fixed-size rectangles and

changes only when the virtual location moves into another rectangle.

For simplicity, scheme #2 is used in this description.

Objects in the cells are displayed as described by the PICT or LABEL elements (or both) in the OBJECTS table. Figure 3 is an example of a display assuming the use of scheme #2.

Objects d1, d2 and d3 are of type OBJDOC (documents); l1 is of type OBJLINK, having the destination coordinates of another cell; and p1 is of type OBJLIFE (another user's avatar). There are also fourteen OBJPATH objects which are displayed (as thick bold lines). The virtual location of the user is (289, 415) represented by the black dot.

Another example of display is the use of an isometric one with more detailed iconic representations for objects. This is shown in Figure 4 and will be described hereafter in more detail with reference to the Common Town user interface.

3.2 Views

3. The concept of "views" is used by servers to keep track of the many displays required by different applications that the user may wish to use. This information is useful for "dynamic updating". In the ideal case, a display falls entirely into the jurisdiction of a server, only a single

view definition is needed. A server maintains a table VIEWS which stores the information of the views that each application uses. Each entry has the following elements:

(VID, UREF, BX, BY, EX, EY, VADDR, TSTAMP)---- VIEWS

Where the elements are:

VID: A unique view identification.

UREF: The UID (from the USERS table) of a user.

BX: The x coordinate of the top-left corner cell of the view.

BY: The y coordinate of the top-left corner cell of the view.

EX: The x coordinate of the bottom-right corner cell of the view.

EY: The y coordinate of the bottom-right corner cell of the view.

VADDR: The network address to which the server can use to communicate with the application which owns the view.

TSTAMP: The time-stamp of the last communication (both to and fro) between the application and the server.

3.3 Display spanning across servers

In the event that a display spans over regions of jurisdiction of several servers (two or more), the display needs the information of the cells from all the OBJECTS tables of these servers. One such case is illustrated in

Figure 5 in which three servers are involved. To make a complete display, at least four views are necessary. They are: region (350, 585, 354, 586) of a server s1, regions (355, 585, 356, 586) and (353, 587, 356, 589) of a server s2 and region (350, 587, 352, 589) of a server s3. In this case, the application needs to define four views across three servers for its single display.

3.4 Stand-In objects

Stand-in objects (OBJSTIN type) have objects as their targets. These targets might be local or foreign. When a stand-in object is contained in the cell of a view, the server needs to keep track of them for dynamic updating. A server keeps a TARGETS table (active-targets table). Each entry has the following components:

(TREF, SREF, VREF)---- TARGETS

Where

TREF: The OID of the target object. Must be local.

SREF: The OID of the stand-in object. Can be local or foreign. In latter case, SID of the server needs to be included.

VREF: The VID of the view which contain the stand-in object. Can be local or foreign. In latter case, SID of server needs to be included.

This is illustrated in Figure 6 in which the viewed portion is (42, 35, 48, 39) and it is defined in the VIEWS table of s1. The stand-in object ▽ in cell (44, 36) needs also to be defined in the TARGETS table of s2 because the target object (t), to which the stand-in object relates, belongs to (ie. is within the jurisdiction of) s2, so only s2 will know when (t) is modified (changed/updated/deleted). Any stand-in object having this object (t) as target needs to register this with s2 so that any change made to (t), is communicated to and reflected by the stand-in object.

3.5 Dynamic updating

An object can be changed by an application. However, in a distributed multi-server and multi-user (many views all over the world on many computer monitors) environment, it is essential to let all users see the latest state of the object. Thus a dynamic update is necessary. For example, if the object is a stock quote info that says "CREAF $16.50" (ie. Creative Labs counter is $16.50 per share now), users all over the world might be (a) viewing the portion of the virtual space with the object and (b) some users might have created a stand-in with the object as a target; any change of the CREAF counter will cause the stock exchange operator (or an automatic application) to update the object. The virtual space will need to inform all users having views containing the object or its stand-in. This is done through the VIEWS table for users of (a) and the TARGETS table for users of

(b).

The single virtual space is maintained by multiple servers and accessed by multiple applications (and users). This makes the virtual space suitable for collaborative computing requiring dynamic update of data.

When elements of an object are changed, the server uses the information in the VIEWS and TARGETS tables to perform a dynamic update. The procedure is as follows:

1. Obtain the coordinates of the changed object.

2. Perform an inclusion test using these coordinates with the VIEWS table and obtain a set of affected views V1.

3. Obtain a set of views V2 which are indirectly affected. This done by looking in the TARGETS table for entries for which TREF is the same as the object's OID.

4. Inform all the applications of the affected views in the union of V1 and V2 by the server contacting the applications using the information in the VIEWS table.

3.6 Resource contention and resolution

"Resource contention" occurs when more than one application updates data in the virtual space at the same time. In a

distributed setting, there are already well known methods to resolve resource contention and these methods can be used effectively with the virtual space.

There are some considerations in the adoption of existing methods, however:

1. In the virtual space, it is assumed that the possibility of resource contention is low because of the fact that the virtual space is large and is accessed by relatively few applications. To avoid the delays caused by lock-and-update strategies, for most types of transactions between applications and servers, a strategy is adopted of letting applications request changes to cells without requesting for locking.

2. When resource contention does occur, the application which causes the contention needs to rewind.

3. Locking is still needed for transactions which are non-rewindable or time-consuming (thus, troublesome to rewind). The latter is usually a practical issue. An example is the dialog box to allow a user to edit the properties of an object, which can be time consuming. Locking is implemented so that the time taken for a user's editing work is not lost because someone has changed the same object during the user's editing.

4. Navigating in the Virtual Space

4.1 Navigating with special objects

To "navigate" in the virtual space means that the user changes his virtual location. He does this so that he changes the view of his display in order to access objects. Some examples of navigation methods are:

1. Using OBJPATH objects. The user can navigate by "following" these objects. The user does this by selecting OBJPATH objects, which are at the edge of the user's view. The application detects that this is the edge of the view, calculates the region for the adjacent view, registers this new view with the server and obtain the cell data from the server for display. A fresh view is displayed with the next adjacent view.

2. Using OBJLINK objects. These objects change the virtual location of the user instantaneously when they are selected. The process of "selecting" is the same as (1) above, but in this case the destination coordinates specified by the DREF element of the OBJLINK object is obtained and the new view is calculated, registered and displayed.

3. Using OBJALINK objects. The OBJALINK object is an aggregate object pointing to an array of OBJLINK objects. When a user selects an OBJALINK object, an application

displays a list, which is the array of OBJLINK objects; the DREF element of the OBJALINK object is the coordinates referring to an array of OBJLINK objects each with its own DREF element which is the coordinates to a possible destination, this list of OBJLINK objects are displayed in a window in textual form (with all the LABEL elements) for user selection, the user can then select from this list the location to which he wishes to navigate.

4.2 Search as Navigation

Search may be used as a means of navigation. A user can search the OBJECTS table for objects with criteria such as matches of attributes such as TYPE and LABEL. The most useful element for search is LABEL. Various algorithms, such as substring matching and close-match search, can be designed to perform search on the LABEL element. Other variations include free-text search on the documents referenced by OBJDOC objects.

The result of a search operation is a list of objects. These are the possible destinations for the user to select and navigate.

4.3 Proximity and Search

The CX and CY elements of the OBJECTS table are useful for a method of search known as "proximity search". When the user

requests for the proximity search operation, he is asked to specify a maximum distance in relation to his virtual location. This distance is used as the cut-off line, surrounding a searching zone, entries in the OBJECTS table with distance (a function of the difference in their CX and CY values) beyond than the cut-off line need not being examined.

Proximity can be used effectively to enhance the speed of the search operation. Instead of searching and returning objects without any order and strategy, nearer objects can be returned and displayed initially to the user. Farther objects returned next and yet farther objects follow. In other words, the search operation is performed in expanding concentric circles (or squares or other regular virtual shape) and results are returned by parts in a stream over time. Assuming that the objects in the virtual space are created and arranged with some logical classification as underlying guidelines, this "expanding concentric search" strategy is effective in reducing the time the user has to wait for the search results to return.

For example, using concentric squares for the expanding concentric search, searching is performed in a sequence of squares q1, q2 ... all of which centre on the user's current virtual location (x, y). The server starts the search with the square region q1 which all four sides are at a distance d away from (x,y). Ie. q1:(x-d, y-d, x+d, y+d). The search

returns all objects which are within q1 using the inclusion test on the OBJECTS table. This is the first search operation. The second search operation is performed with the square q2:(x-2d, y-2d, x+2d, y+2d) but exclude q1 (a normal SQL database can handle this type of inclusion search quite easily). Theoretically, this sequence can continue indefinitely but in practice there are two conditions for cut-off: (a) the sequence can be cut-off if a search with say, qn, reaches the edge of the virtual space, ie. there are no server managing regions outside of qn (b) the user/application can specify a cut-off distance.

The time the user has to wait for the search results to return is reduced because the search operation may be partitioned: the search with q1 can be performed and while the results are returned to the user/application, q2 is being searched; and so on. The wait time for the first search results is thus reduced and nearer objects are returned first.

4.4 Searching Across Servers

Searching across servers may be performed using the strategies discussed.

For example, using a concentric square search strategy, for each search with a concentric square qn:(x-n.d, y-n.d, x+n.d, y+n.d), the SREGIONS table is used to examine which are the

appropriate servers to handle the search. Each entry r in this table is tested for inclusion with the four corners of qn, which are (x-n.d, y-n.d), (x-n.d, y+n.d), x+n.d, y-n.d) and (x+n.d, y+n.d). If any corner is within r, the corresponding server s has to perform a search operation with the region qn. A search operation with any single qn can potentially involved more than a server (ie. qn cuts across servers' boundaries), in which case the search results from all the involving servers are combined and returned.

Using the proximity search feature noted above, a practical search strategies may be employed by letting the user specify the distance to cut-off. Using the SREGIONS table, servers beyond the cut-off distance need not be searched.

5. Default Objects and Population Functions

Very often in practice, it is desirable to have default objects in cells so that the virtual space is populated by objects and does not look empty. For example, one might want to sprinkle an object OBJTREE, having an icon of a tree in the virtual space so that the entire space looks sparsely vegetated. On the other hand, creation of these objects as entries in the OBJECTS table is not desired because the objects take up storage space.

Special mathematical functions can be use to achieve this, here referred to as "populating functions". A populating

function takes the form:

$$f(x, y) \rightarrow \{0, 1\}$$

Where x and y are the coordinates of a cell. The output is Boolean, which indicates the existence of the object in the cell.

An example is a function which populates the virtual space with virtual treesof the form:

g(x, y) → {1: (x+x.y mod 17)>14, 0: otherwise}

where the function "mod" is the modulo function (ie. the result of j mod k is the remainder of j divides by k).

These objects are not created as entries in the OBJECTS table. Due to their on-demand and infinite nature, populating functions are an inexpensive means to generate default objects.

These populating functions are called before data is retrieved from the OBJECTS table. There can be more than one populating function and multiple default objects can exist for a cell. Actual objects of a cell might or might not override these default objects.

For example: a cell is computed to contain a default object which is a virtual tree, the OBJECTS table shows that cell contain an object of type OBJPATH. In this case, the virtual

tree is ignored by the application and only the OBJPATH object is shown in the display. To ignore or not is application dependent. The cell might at the same time contain a virtual tree and a virtual river and display the OBJPATH as a road spanning over the river (ie. a road bridge).

6. Application Program Interface (API)

The conceptual structure of the virtual space has been discussed in sections 1-5. In order for the structure to interface with users, it is necessary for an application program interface (API) to be provided. The API comprises a plurality of sub-routines which can be conveniently divided into three groups, end-user, access management and server management. Although the nature of such sub-routines and their desirability would be clearly apparent to the man skilled in the art, a brief resume of desirable sub-routines is provided hereinafter.

6.1 End-User

These are the transactions defined between an end-user application and a server.

6.1.1 Create object: Add an entry in the OBJECTS table.

6.1.2 Modify object: Modify the elements of an entry in the OBJECTS table.

6.1.3 Move object: Modify the CX and CY elements of an entry in the OBJECTS table.

6.1.4 Copy object: Copy an entry in the in the OBJECTS table to another entry. The destination OBJECTS table can be local or foreign.

6.1.5 Get object: Retrieve an entry in the OBJECTS table.

6.1.6 Delete object: Delete an entry in the OBJECTS table.

6.1.7 Create view: Add an entry in the VIEWS table.

6.1.8 Delete view: Delete an entry in the VIEWS table.

6.1.9 Move view: Modify the region of an entry in the VIEWS table.

6.1.10 Create active-target: Add an entry in the TARGETS table.

6.1.11 Delete active-target: Delete an entry in the TARGETS table.

6.1.12 Sign-in: Identify the user of the application to the server by providing a UNAME and PASSWD. There is no sign-out transaction; when the application terminates, corresponding views are deleted.

6.1.13 Search object: Search for objects in the OBJECTS table.

6.2 Access management

Access management applications are those that manage user,

group, zone and badge information of a server. These are the transactions defined between the server and an access management application:

6.2.1 Create user: Add an entry in the USERS table. Can be either UUSER or UGROUP type (i.e. user or group).

6.2.2 Modify user: Modify the elements of an entry in the USERS table.

6.2.3 Delete user: Delete an entry in the USERS table.

6.2.4 Create membership: Add an entry in the GMEMBERS table.

6.2.5 Delete membership: Delete an entry in the GMEMBERS table.

6.2.6 Create zone: Add an entry in the ZONES table.

6.2.7 Modify zone: Modify the elements of an entry in the ZONES table.

6.2.8 Delete zone: Delete an entry in the ZONES table.

6.2.9 Create badge: Add an entry in the BADGES table.

6.2.10 Modify badge: Modify the elements of an entry in the BADGES table.

6.2.11 Delete badge: Delete an entry in the BADGES table.

6.2.12 Grant badge: Add an entry in the UBADGES table.

6.2.13 Withdraw badge: Delete an entry in the UBADGES table.

6.2.14 Identify the user of the application to the server by providing a UNAME and PASSWD. There is no need for a sign-out transaction.

6.3 Server management

Server management applications are those that manage the information servers. The applications are found usually in the Land Bureau computer system. These are the transactions defined between the Land Bureau system and a server management application:

6.3.1 Create server region: Add an entry in the SREGIONS table.

6.3.2 Modify server region: Modify the elements of an entry in the SREGIONS table.

6.3.3 Delete server region: Delete an entry in the SREGIONS table.

6.3.4 Create server: Add an entry in the SERVERS table.

6.3.5 Modify server: Modify the elements of an entry in the SERVERS table.

6.3.6 Delete server: Delete an entry in the SERVERS table.

6.3.7 Identify the user of the application to the server by providing a UNAME and PASSWD. There is no need for a sign-out transaction.

7. Metaphors

A metaphor is a representation scheme for the virtual space at the user interface. There are many metaphors suitable for the user interface. One of these is the "town" metaphor. In this metaphor, OBJDOC objects are shown as buildings and

OBJPATH objects are shown as roads. OBJLINK objects are bus stop and OBJALINK are subway stations. The searching tool can be a Taxi and OBJLIFE objects are users' avatars. This will be described hereinafter with reference to a specific application termed Common Town.

There are other possible metaphors such as "tunnels and caverns" or "corridors and rooms" which are similar to the town metaphor. Another is the "infinite desktop" metaphor, which provide a single shared desktop to distributed users and servers.

There are many applications for the virtual space. One is to use the virtual space together with the World-Wide-Web in which OBJDOC objects are Web documents. The collaboration-enabling characteristics of the virtual space are clear. Stand-in objects are also very useful in the Web environment. They can provide "live" information changed constantly by entities providing services such as news, stock-quotes and weather reports. The stand-in objects are informed and refreshed through the dynamic updating feature of the virtual space. Furthermore, the user can create stand-in objects and put them close together for convenience.

8. Common Town

An example of the use of the virtual space, hereinafter referred to as CommonTown will now be described. CommonTown

is a software program providing a virtual landscape that allows the creation and operation of online virtual communities over the Internet. Users all over the Internet can access the objects placed in this virtual landscape and can create objects to be seen and accessed by other users. Objects are represented as buildings, roads and automobiles, things that are borrowed from the real world. This creates a "town" metaphor such that users can feel a strong sense of community when they are using CommonTown and perform tasks in a familiar setting.

Common Town provides many features useful for virtual communities. It enables users to see one another as avatars taking the form of people, animals or automobiles which can traverse the landscape. Clicking on these avatars lets a user send instant messages or email to the avatar's owner. Users can own and share land and create web pages at the click of a button. Common Town shows buildings (OBJDOC objects) which are web pages, chat rooms and forums (newsgroups). The users can create these buildings by simple drag-and-drop from an icon menu into the landscape and the building can then be edited.

Aspects of Common Town will now be described with reference to the general features of the virtual space described previously with particular reference to the client and server implementations, the various objects which can populate the landscape, items associated with these objects in particular

web pages, landscape construction tools, specific ways of navigating in the landscape, community-related features and the client-server model.

This description of Common Town provides an indication to the skilled man of the kind of environments that can be developed using the virtual space described above. In this respect, the design of software to perform the functions described in Common Town below are well within the capabilities of those skilled in the art once aware of the contents of the present application.

The client is implemented as a computer program known as the CommonTown Viewer which operates inside a Web-browser such as Netscape Navigator or Internet Explorer and which performs the functions noted in paragraph 3.2. An example of the display provided by the viewer is shown in Fig. 4. The program only needs limited functions and is therefore lightweight and can be distributed over the Internet quickly and installed by users themselves.

The CommonTown Viewer is like a "scope" that looks into the virtual landscape. This is shown in Figure 6. As noted in Section 3 above, a fixed orthogonal view is used. Objects displayed in the Viewer looks 3-dimensional but in fact they are 2-dimensional bitmaps pre-rendered at a common fixed viewing angle. As such, the client computer does not need 3D rendering capabilities.

The grid described in Section 1.1 above forms the CommonTown landscape, dividing the landscape into cells. The user uses the Viewer to peek into this vast landscape one rectangle at a time. The Viewer displays a rectangle of 7x6 cells.

The objects defined in CommonTown, with reference to the general types mentioned in Section 1.2 above, are:

Object	Type	Description
Road	OBJPATH	Roads link up different parts of the land for navigation purpose and for providing loose visual organisation clues.
Building	OBJDOC	Buildings are "data holders", their WebPage property can contain either a URL to an existing web page or a web page maintained by Common Town
Signboard	OBJSIGN	Signboards are objects which contain text labels that are displayed prominently. Usually they are used to mark neighbourhoods or parts of the landscape. Signboards are very useful as search targets. Eg. "Home of the National Lottery".
Pointer	OBJPTR	Points are signs that gives users directions to neighbourhoods. Eg. "Go left to News Zone".
Bust-stop	OBJLINK	Bus-stops are objects that transport users instantaneously to other locations on the land. Eg. "To Shopping Malls".
Subway-stop	OBJALINK	Sub-way stops are objects that store a list of destinations. Usually these destinations are prominent neighbourhoods of the landscape. A user clicks on a subway-stop can choose from the list of destinations to be transported to.

Tree	OBJTREE	A default object. Trees are provided merely as decoration of the landscape.
Avatar	OBJLIFE	Avatars are net-presence of users. Every user has a single avatar in the entire CommonTown (even in multi-servers environment).
Stand-in	OBJSTIN	Stand-ins are links that refer to other objects. Only two types of objects can have stand-ins: house and avatar

Examples of objects as displayed are shown in Fig. 4 such as Roads 200, Buildings 300, Signboard 400, Bu-stop 450, Pointers 500 and Trees 600.

The CommonTown Viewer provides the End User functionalities through the use of the Application Program Interface mentioned in Section 6.1 as tools for users to construct on the landscape. The user can create objects and modify their properties. Object creation is accomplished by selecting tools from a palette 100 to the right of the Viewer as shown in Fig. 4. For example, a signboard tool can be selected by clicking on the appropriate icon with a mouse to display a dialog box for input of information to create a signboard object OBJSIGN in a cell on the landscape. A Building (Object OBJDOC) can similarly be created using another tool and so on.

There are three general editing tools for modifying objects properties and the layout of the landscape: Renovator 110

(see 6.1.2 above), Smasher 120 (see 6.1.6 above) and Mover 130 (See 6.1.3 above). The Renovator tool allows the properties such as the name and behaviour of an object to be modified. The Smasher tool demolishes (or removes) objects placed earlier on the landscape. The Mover tool moves objects on the landscape from one cell to another.

A feature of the CommonTown Viewer is the ease with which the user can create web pages. A document object (OBJDOC) represented by a Building icon, can be a web page and can be edited directly by invoking a web page editor (depending on the web-browser used, it will be Netscape Composer or MS FrontPage Express). The integrity of a CommonTown web page is maintained by the server by restricting editing access and resolving access conflicts using the methods described in Section 2 above.

CommonTown web pages are as situated on a common workspace and are, in principle, accessible to users located anywhere on the global Internet.

Objects are displayed as icons in the Viewer. An icon is either a built-in picture of which examples are shown in Figure 4, an external picture (a GIF file specified using a URL), or an animation. The CommonTown Viewer may have a collection of built-in pictures and animations loaded into users' computers during the installation process. This will reduce the runtime bandwidth requirement and contributes to

efficient online performance.

The "town" metaphor provides for rich navigation methods borrowed from the real-world. This includes the simplest navigation using roads to concepts such as subway systems.

The most basic navigation method in the CommonTown virtual landscape is the use of roads (OBJPATH objects). If the road crosses the edge of the CommonTown Viewer, a little arrow-shape icon is shown out the edge. By clicking on this icon, the Viewer pans the display to the adjacent map-square. This method of navigation prevents the user from reaching map-squares which have no roads. This may at first sight appear to be a disadvantage, but this restriction prevents the user from panning freely over the landscape which causes disorientation. Roads serves as important landmarks for navigation and if the user follows roads when navigating, he has a more concrete mental picture of the virtual landscape.

The town metaphor provides a good equivalence for the search tools mentioned in sections 4.2 - 4.4 and a move function, namely Taxi. The Taxi tool 140 which provides the functionalities mentioned in S6.1.14 and S6.1.9, when selected, shows a Taxi dialog box, which asks for a destination. The exact name of the destination need not be given, the Taxi tool can accept a searchable sub-string or other definition, for example object type, and will provide suggestions by displaying objects whose labels match the sub-

string partially. The user can then select one of these objects and be transported to the map-square it is located.

The Taxi tool in the CommonTown landscape may provide the proximity search feature discussed in Section 4.3. Since any two points in the CommonTown landscape have a definite two-dimensional relationship, the Taxi tool allows the user to limited the search to the vicinity the user. This is done by a distance slide-bar in a Taxi dialog box which can be used to specify the dimensions of search as discussed in Section 4.3.

A Bus-stop 450 is an OBJLINK object which provides another way of navigation in the CommonTown landscape. A bus-stop object, when clicked, transports the user from one location to another. Bus-stops may be constructed using a tool 150 similar to those provided on palette 100.

Subway-stops are objects which provide an even more organized way of navigation. A subway line is an OBJALINK aggregate object made up of a plurality of locations (subway-stops) which can be placed anywhere in the landscape. When the user clicks any of these subway-stops (not shown in Fig. 4), a list of subway-stops is displayed prompting the user to select a subway-stop on this subway-line to be transported to.

Subway lines can be used to organize navigation for multiple

levels. The overall landscape of a CommonTown server, for example, can be divided into various districts for commerce, games, education, social and news services. A major subway line can tie up these districts by providing stops in the hearts of these districts. Each district can have its own thematic subway line to serve its several neighbourhoods. The district of games services, for example, can have a minor subway line linking up its neighbourhoods for board games, first-person shooters, puzzles etc. The major and minor subway lines can meet at a subway-stop serving as an exchange.

Subway lines need not to be a list strictly. They can be nested as described in Section 1.3.

The last navigation tool is a Homeland Tool 160. This is the tool marked with "home" on the right side of the Viewer. This is simply the tool to transport the user to the land specified as his homeland where his avatar is located and uses the move view application (S6.1.9) to "return" the user to the location OBJREF specified in the user's entry in the USERS table.

CommonTown users register with CommonTown servers to obtain online identities as discussed in Section 2 above. Each user has to register only once with a server by providing an e-mail id and other personal particulars such as name, etc. to provide the information necessary for the USERS table. The

registration is valid globally across other CommonTown servers.

To start a session, the user signs in with the server with which he is registered using the Sign-In subroutine (S6.1.12). The user needs to provide his username and a password in order to assume his registered identity. This process identifies him to the server and his access rights are initialized.

An avatar is an OBJLIFE object and is the virtual manifestation of a user representing the user on the CommonTown landscape. One avatar object may be insufficient and two objects are preferred: the shell (OBJLIFE object) to serve as the static representation and the soul (OBJSTIN object) as the roaming counterpart, which is a stand-in object. The shell occupies a cell in the user's homeland. This is a convenient reference for other users who need to communicate with the user via emails or instant messages. The soul strolls about and appears in the map-square where the user is currently viewing. Other users viewing the same map-square can see the soul of the user represented as a stroller object (e.g. an automobile, a cow or a blimp). Similarly, email and instant messages can be sent to the soul. If a user leaves CommonTown after a session, the soul will return to the shell.

The avatar implementation has the capability to show another

user who is browsing the user's web page or a web site, providing this feature in a visual and intuitive way.

CommonTown gives the impression that there is a single common piece of land across CommonTown host servers. A central controlling web site, the (CommonTown) Land Bureau (CLB) mentioned in Section 1.1, is responsible for carving out mutually exclusive pieces of land for these servers using the Access Management Subroutines of S6.3. Each CommonTown server manages a piece of land (region) assigned by the CLB. Each server has control of the users registered with it and the access right mechanism within the land's boundary. This land is under the server's jurisdiction. This scheme is scalable, providing unlimited land to be added and managed by new servers outwardly in the four compass directions. This scheme also gives the user a good sense of directionness and put him in relation to objects in the landscape when he is browsing the landscape.

The CLB periodically distributes any change in geography of the landscape (i.e. servers' jurisdiction information) to every server in the form of the SREGIONS and SERVERS tables. During browsing, a server uses the information of this look-up table to transfer a user who moves out of its jurisdiction to the appropriate server that manages the adjacent piece of land.

CommonTown uses the HTTP connectionless interface to

communicate between clients and servers. The HTTP interface, due to the WWW, is a prevalent infrastructure present on most server and client computers and this allows the installation

CommonTown uses the concept of badges described in Section 2 for access control of land and objects. Users are given badges individually or as a result of the groups they joined.

A CommonTown server issues basic badges to new users. New badges can be issued to a user and existing badges can be removed over time. If the server has many users and many badges, the management of badges can be alleviated using the concept of "groups" described in Section 2.1. A group has a list of badges associated with it. A user who belongs to a group inherits all its badges.

CommonTown server administrators are provided with management applications (Section 6.2) to manage users, badges, groups and land.

The embodiment of the virtual space described is not to be construed as limitative. For example, although the virtual space has been illustrated as a two-dimensional coordinate grid, this could be in another form using another coordinate system and may be in more than two dimensions. Furthermore, in the described embodiment and in the claims, reference is made to a "table" as means for storing information. It is to be understood that the term "table" where it appears in the

claims should be construed in an abstract sense as referring to a depository of information in a form that can be retrieved.

CLAIMS

1. A virtual space extending over a plurality of servers, the space comprising a grid defining a plurality of cells at coordinate points of the grid and means for associating each cell with at least one object and the grid comprises a plurality of regions, each region being disposed on a respective server.

2. A virtual space as claimed in claim 1 wherein each server has jurisdiction over its region.

3. A virtual space as claimed in Claim 1 or claim 2 wherein each region is connected to at least one other region at a common boundary.

4. A virtual space as claimed in claim 3 wherein the connected regions are coordinately contiguous.

5. A virtual space as claimed in any one of the preceding claims wherein a said object comprises a document.

6. A virtual space as claimed in any one of the preceding claims wherein a said object connects different parts of the virtual space.

7. A virtual space as claimed in any one of the preceding claims wherein a said object references the coordinates of

another cell.

8. A virtual space as claimed in any one of the preceding claims wherein a said object defines a user of the virtual space.

9. A virtual space as claimed in any one of the preceding claims wherein a said object references another object in the virtual space.

10. A virtual space as claimed in any one of the preceding claims wherein each object and the location thereof is defined in an object table.

11. A virtual space as claimed in claim 10 wherein the object table is implemented as a sparse matrix.

12. A virtual space as claimed in claim 10 or claim 11 wherein each server has a said object table defining objects existing in cells in the region of the server.

13. A virtual space as claimed in any one of the preceding claims further comprising means for referencing a plurality of objects together.

14. A virtual space as claimed in claim 13 wherein said means comprises a defined array of said plurality of objects.

15. A virtual space as claimed in claim 13 or claim 14 wherein at least one of said plurality of objects references a further plurality of objects.

16. A virtual space as claimed in any one of the preceding claims further comprising a user table which identifies users of the space.

17. A virtual space as claimed in claim 16 wherein each server has a user table identifying users having home locations in the region of the server.

18. A virtual space as claimed in claim 16 or claim 17 wherein each entry in the or each user table includes an identification of the user in the table, a name for the user, a password associated with the user and a contact address.

19. A virtual space as claimed in claim 18 wherein the contact address is an E-Mail address.

20. A virtual space as claimed in any one of the preceding claims further comprising at least one group table, identifying a group of users of the space.

21. A virtual space as claimed in claim 20 wherein each server has a group table identifying a group of users having home locations in the region of the server.

22. A virtual space as claimed in claim 20 or claim 21 wherein each entry in the or each group table includes a definition of a group and an identification of a user that is part of that group.

23. A virtual space as claimed in any one of the preceding claims further comprising at least one table of access control zones, each zone having access requirements associated therewith.

24. A virtual space as claimed in claim 23 wherein each server has a said table of access zones, each zone being disposed within the region of each server.

25. A virtual space as claimed in claims 23 or claim 24 further comprising a table of badges, each badge having access rights associated therewith, access to a zone being granted when the access rights of a holder of a badge meet the access requirements of a zone.

26. A virtual space as claimed in claim 25 further comprising a rights table of users and the badges held by each user.

27. A virtual space as claimed in claim 25 or claim 26 wherein each server has a said rights table containing badges held by users having home locations in the region of the server.

28. A virtual space as claimed in claim 27 wherein the said rights table of each server contains badges for zones in the region of the server.

29. A virtual space as claimed in claim 28 wherein the rights table of each server contains badges held by users having home locations in regions of other servers.

30. A virtual space as claimed in any one of the preceding claims further comprising means for displaying a portion of the grid and representations of objects contained in cells in said portion.

31. A virtual space as claimed in claim 30 wherein the portion contains the virtual location of the user.

32. A virtual space as claimed in claim 31 wherein the virtual location of the user is at the centre of the displayed portion.

33. A virtual space as claimed in claim 31 wherein the portions are non-overlapping parts of the grid.

34. A virtual space as claimed in claim 33 wherein a corner of the displayed portion has grid coordinates which are integer divisible by the cell dimensions of the portion.

35. A virtual space as claimed in any one of claims 30 to 34 wherein, when a portion extends over more than one region, the display means obtains information concerning the objects within the part of each region to be displayed from the region's server and displays the parts together as a single display.

36. A virtual space as claimed in claim 9 or any claim dependent thereon further comprising a table of referring objects and corresponding referred objects

37. A virtual space as claimed in claim 36 wherein each server has a table of referring objects and corresponding referred objects in respect of referring objects within its region.

38. A virtual space as claimed in any one of the preceding claims further comprising means for updating references to and views of objects.

39. A virtual space as claimed in claim 38 wherein each server updates references and views within its region.

40. A virtual space as claimed in any one of the preceding claims further comprising means for changing the virtual location of the user.

41. A virtual space as claimed in claim 40 wherein said

means comprises using an object to change the virtual location.

42. A virtual space as claimed in claim 41 wherein the object is selected from an object as claimed in claim 6 or claim 7.

43. A virtual space as claimed in claim 40 wherein said means comprises means for moving to the location of a selected object by searching for objects having at least one attribute and selecting the object from the search results.

44. A virtual space as claimed in claim 40 wherein said means comprises means for specifying a zone, displaying references to objects associated with cells within that zone, selecting a referenced object and moving to the virtual location of that object.

45. A virtual space as claimed in claim 44 wherein the zone includes the virtual position of the user.

46. A virtual space as claimed in claim 45 wherein the zone is centred on the virtual position of the user.

47. A virtual space as claimed in any one of claims 44 to 46 wherein the zone is a regular geometric virtual shape.

48. A virtual space as claimed in any one of claims 44 to 47 wherein the specifying means specifies a plurality of zones of expanding size.

49. A virtual space as claimed in claim 48 wherein the specifying means displays object references within one zone before displaying object references within the next larger zone.

50. A virtual space as claimed in any one of the preceding claims further comprising means for generating default objects in cells of the grid.

51. A virtual space as claimed in claim 50 wherein said means comprises using a populating function of the form:

$$f(x,y) \rightarrow \{0,1\}$$

where x and y are the coordinates of the cell in which the default object is to be disposed.

52. A virtual space as claimed in any one of the preceding claims having a viewing metaphor.

53. A virtual space as claimed in claim 52 wherein the metaphor is that of a community.

54. A virtual space as claimed in claim 53 wherein the metaphor is a town metaphor in which an object is represented by a building for an object as claimed in claim 6, by a road

for an object as claimed in claim 7, by public transport for an object as claimed in claim 8 and by an avatar of the user for an object as claimed in claim 9.

55. A virtual space as claimed in any one of the preceding claims wherein said plurality of servers are interconnected via the internet.

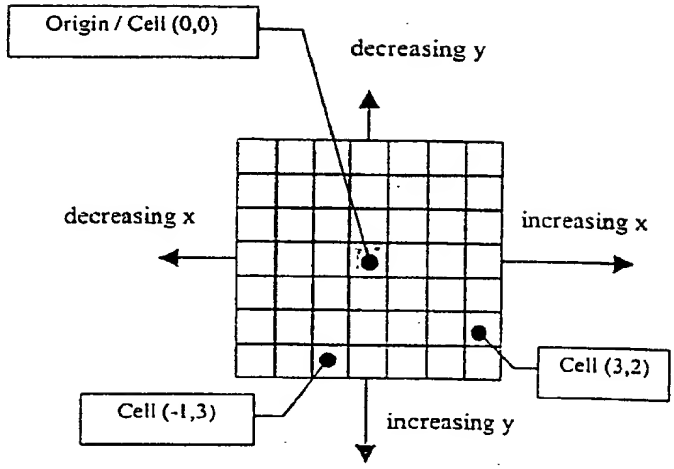


FIGURE 1

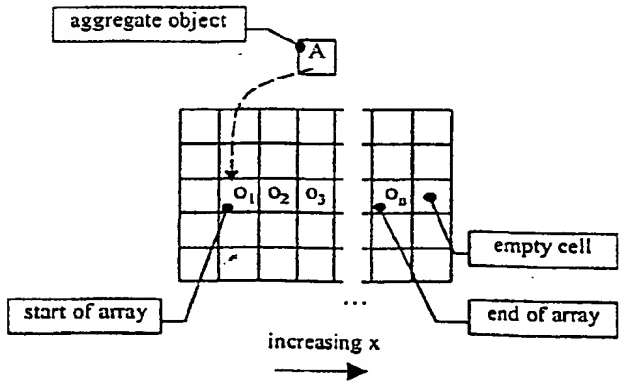


FIGURE 2

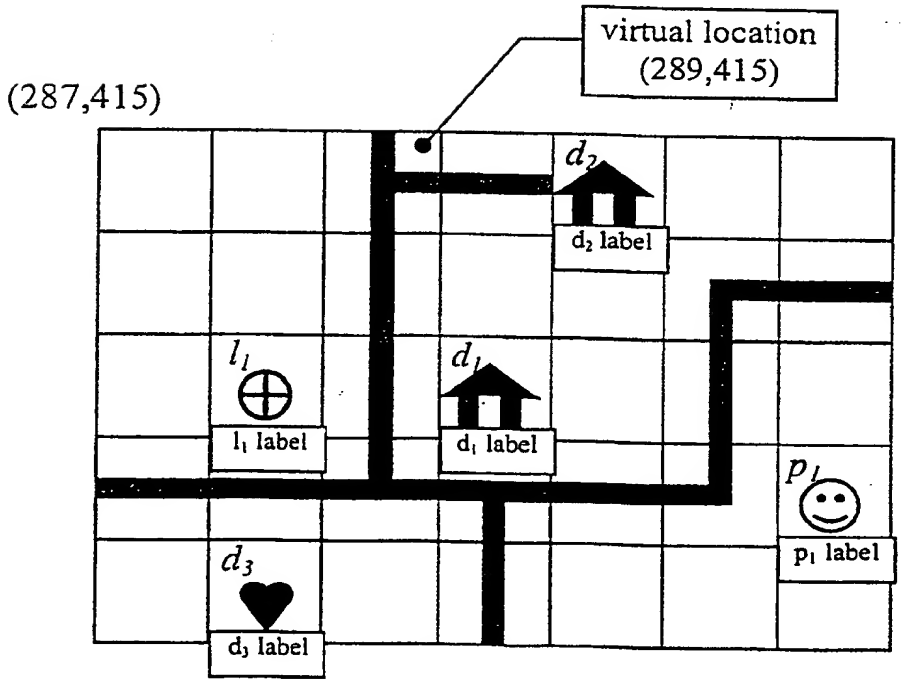


FIGURE 3

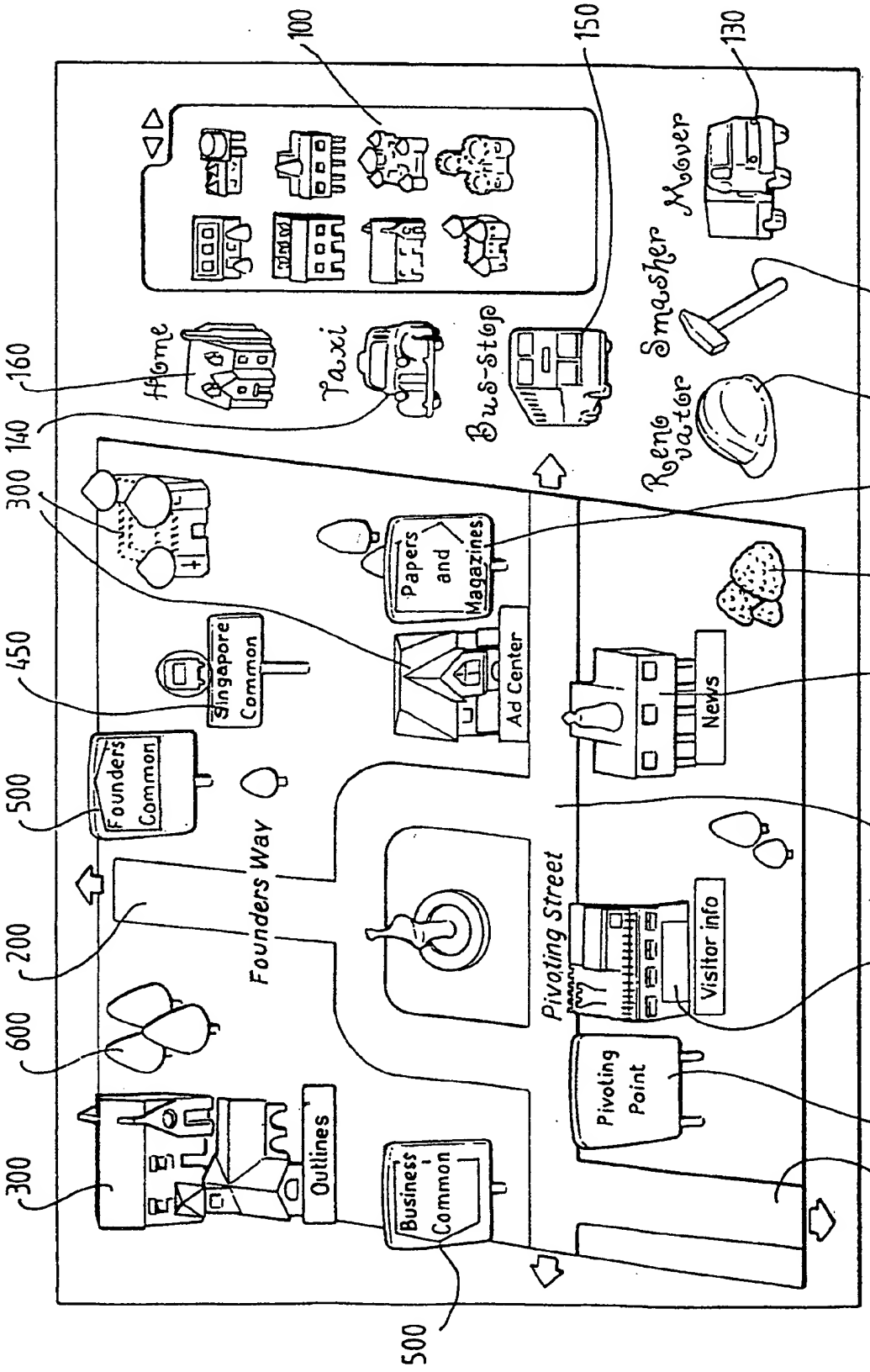

100
150
130
Mover
Smasher
120
Home
Taxi
Bus-stop
160
140
110
Renovator
300
500
Papers and Magazines
600
450
Singapore Common
Ad Center
News
300
500
Founders Common
Founders Way
200
Pivoting Street
200
600
Visitor info
300
Outlines
Pivoting Point
400
300
Business Common
200
500

FIGURE 4

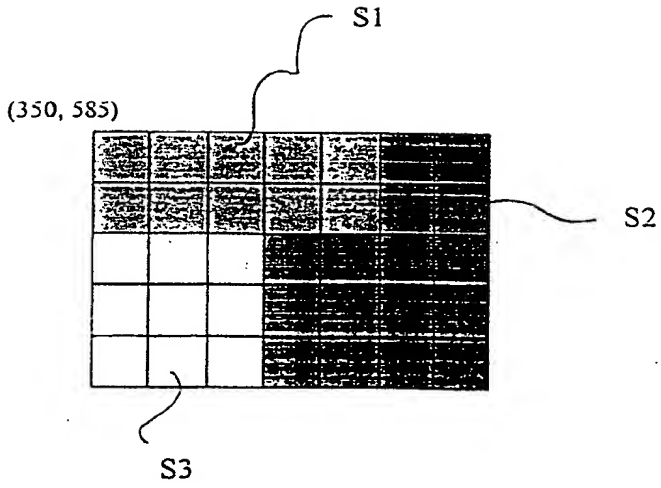


FIGURE 5

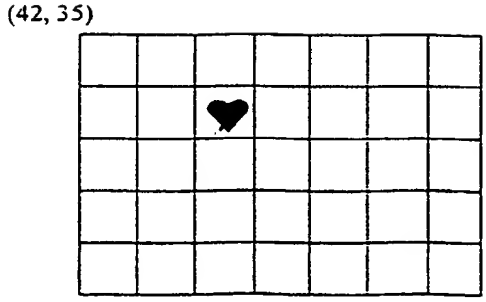
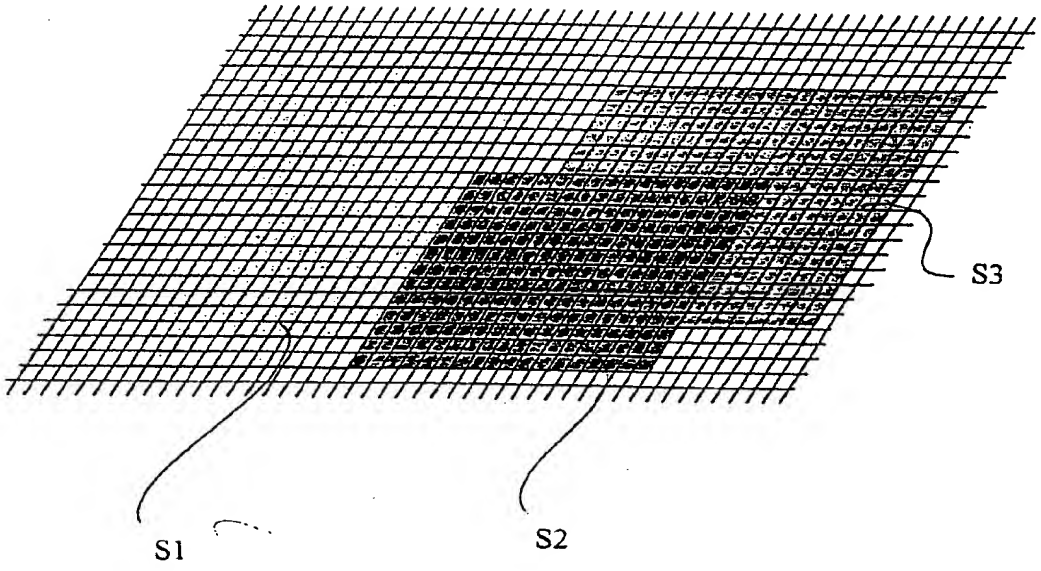


FIGURE 6

FIGURE 7



INTERNATIONAL SEARCH REPORT

International application No.
PCT/SG 99/00100

A. CLASSIFICATION OF SUBJECT MATTER

IPC[6]: G06 F 17/30

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC[6]: G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPODOC, WPI, PAJ, ACM digital Library

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0753 836 A (SONY), 15.January 1997 (15.01.97) abstract, fig. 1,9,10-11.	1,2,5-10,12,16, 18-20,30-32,38-42, 52-55
A	BENFORD S. et al. 'Understanding and Constructing Shared Spaces with Mixed-Reality Boundaries', ACM Transactions of Computer-Human Interaction, Vol.5, No.3, September 1998, pp. 185-223. totality; especially 2.5 Spatiality, 4. Mixed-reality Boundaries, 5. the Internet Foyer-a demonstration of a mixed-reality boundary	1,3,5,6,30,31,52
A	BLUMENOW W. et al. 'The Process Agent Model and Message Passing in a Distributed Processing VR System'. Proceedings of the ACM symposium on Virtual reality software and technology, September 15-17,1997, Lausanne Switzerland. totality; 3. Conceptual Virtual Reality System Model	2,16,30,31,38,39
A	US 5,659,691 A (PRETTEGIANI), 19. August 1997 (19.08.97), abstract.	1

☒ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:
„A" document defining the general state of the art which is not considered to be of particular relevance
„E" earlier application or patent but published on or after the international filing date
„L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
„O" document referring to an oral disclosure, use, exhibition or other means
„P" document published prior to the international filing date but later than the priority date claimed
„T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
„X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
„Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
„&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
27 March 200 (27.03.00)	29 March 2000 (29.03.00)
Name and mailing adress of the ISA/AT Austrian Patent Office Kohlmarkt 8-10; A-1014 Vienna Facsimile No. 1/53424/535	Authorized officer Fastenbauer Telephone No. 1/53424/447

Form PCT/ISA/210 (second sheet) (July 1998)

INTERNATIONAL SEARCH REPORT

International application No.
PCT/SG99/00100

C (Continuati n). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	BROLL Wolfgang, 'Populating the Internet: supporting multiple users and shared applications with VRML '. Proceedings of the second symposium on Virtual reality modeling language,February 24 - 26, 1997, Monterey, CA USA , pp.33-40. ---	1,3,4,16-19,23,25-29, 40-41

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.
PCT/SG 99/00100

Patent document cited in search report			Publication date	Patent family member(s)			Publication date
EP	A2	753836	15-01-1997	CA	AA	2180891	13-01-1997
EP	A3	753836	29-01-1997	JP	A2	9081781	28-03-1997
				US	A	5956038	21-09-1999
US	A	5659691	19-08-1997	AU	A1	77366/94	10-04-1995
				AU	B2	687888	05-03-1998
				CA	AA	2172535	30-03-1995
				EP	A1	721614	17-07-1996
				JP	T2	9503082	25-03-1997
				NZ	A	273893	27-05-1998
				NZ	A	330003	28-10-1999
				WO	A1	9508793	30-03-1995
				US	A	5950202	07-09-1999